

**This Page Is Inserted by IFW Operations
and is not a part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- **BLACK BORDERS**
- **TEXT CUT OFF AT TOP, BOTTOM OR SIDES**
- **FADED TEXT**
- **ILLEGIBLE TEXT**
- **SKEWED/SLANTED IMAGES**
- **COLORED PHOTOS**
- **BLACK OR VERY BLACK AND WHITE DARK PHOTOS**
- **GRAY SCALE DOCUMENTS**

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.**

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-198643

(43) 公開日 平成10年(1998) 7月31日

(51) Int.Cl.⁶
G 0 6 F 15/16
9/46

識別記号
3 7 0
3 6 0

F I
G 0 6 F 15/16
9/46
15/16

3 7 0 N
3 6 0 C
4 2 0 J

審査請求 未請求 請求項の数 6 O L (全 11 頁)

(21) 出願番号 特願平9-2061

(22) 出願日 平成9年(1997) 1月9日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(71) 出願人 000153443

株式会社日立情報制御システム

茨城県日立市大みか町5丁目2番1号

(72) 発明者 加藤 裕昭

茨城県日立市大みか町五丁目2番1号 株式会社日立情報制御システム内

(72) 発明者 仲田 智

茨城県日立市大みか町五丁目2番1号 株式会社日立情報制御システム内

(74) 代理人 弁理士 高崎 芳紘

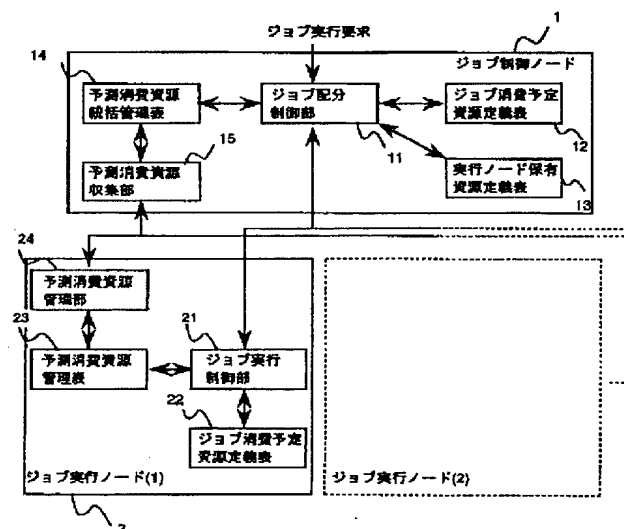
最終頁に続く

(54) 【発明の名称】 分散計算機システム

(57) 【要約】

【課題】 ジョブを配分するに際し、その配分の判断が計算機資源のジョブ起動時点における情報のみによって行われており、時間の経過に伴い計算機上で実行されている複数のジョブによって資源が消費され、資源の不足により計算機を停止させる可能性がある。

【解決手段】 ジョブ消費予定資源定義表12に予め各ジョブの消費予定資源が定義されている。ジョブの実行時に、ジョブ配分制御部11、ジョブ実行制御部21、ジョブ消費予定資源定義表22、予測消費資源管理表23を用い、ジョブ消費予定資源定義表12の消費予定資源及び予測消費資源情報を基に、前記ジョブを実行するジョブ実行ノード2において必要とされる将来の消費予定資源が前記ノードの有する資源をこえない様にジョブを配分し、稼働停止を防止する。



【特許請求の範囲】

【請求項1】 複数の計算機から構成された分散計算機システムにおいて、実行対象のジョブの消費予定資源を予め定義するジョブ消費予定資源定義手段と、前記ジョブの実行時に、そのジョブを実行するノード上で必要とされる将来の消費予定資源が前記ノードの有する資源を超えない様にジョブの配分を行うジョブ配分手段とを具備することを特徴とする分散計算機システム。

【請求項2】 前記ジョブ配分手段は、前記ジョブ消費予定資源定義手段で定義されたジョブの消費予定資源と予測消費資源統括管理手段で管理されている前記ノード上の予測消費資源との和がノード保有資源定義手段で定義された前記ノード上の保有資源を越えないようにジョブを配分することを特徴とする請求項1記載の分散計算機システム。

【請求項3】 前記ジョブ配分手段は、前記ジョブ配分に加え、前記ジョブ消費予定資源定義手段で定義されたジョブの消費予定資源と予測消費資源統括管理手段で管理されている前記ノード上の予測消費資源との和に対する前記ノード上の予測消費資源の比率が最小になり、かつ前記和がノード保有資源定義手段で定義された前記ノード上の保有資源を越えないようにジョブを配分することを特徴とする請求項2記載の分散計算機システム。

【請求項4】 前記ジョブ配分手段は、スキップされたジョブの起動回数が予め設定した値を越える時、前記スキップされたジョブを優先的に実行させることを特徴とする請求項1記載の分散計算機システム。

【請求項5】 前記ノードに代えて、データベースサーバを用いることを特徴とする請求項1記載の分散計算機システム。

【請求項6】 前記ノード又は前記データ処理サーバは、ジョブ実行要求に応じて消費予定資源値又は記憶資源アカウント値を取得し、この値を予測消費資源管理手段又は記憶資源アカウントテーブルに加算してジョブを起動し、その実行後に前記予測消費資源管理手段から消費予定資源値を減算又は前記記憶資源アカウントテーブルから前記記憶資源アカウント値を減算する手段を具備することを特徴とする請求項1又は請求項5記載の分散計算機システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、複数の計算機ノードで構成された分散計算機システムに関するものである。

【0002】

【従来の技術】 近年の計算機システムは、処理を高速化する要求、処理データ量の増大、及び計算機間を接続するネットワークの発展により、複数の計算機をネットワークで接続した分散計算機システムを構成するケースが多くなってきている。分散計算機システム上でジョブを

実行する場合、計算機資源（1群の仕事を計算機システムで実行するために必要なハードウェア及びソフトウェア）を有効に活用するためには、個々の計算機の資源消費状態を管理しながら各計算機に最適にジョブを配分する必要がある。

【0003】 従来、分散計算機システムにおけるジョブ配分は、ジョブの新規起動時に、（i）各計算機のジョブ配分時点における資源消費状態、及び（ii）予め定められたポリシーに基づいてジョブの配分を決定していた。

【0004】

【発明が解決しようとする課題】 しかし、従来の分散計算機システムにおけるジョブ配分は、ジョブの制御を行うために管理する各計算機上の資源情報が、刻々変化する計算機上の資源のある時間断面におけるデータに基づいている。このため、ジョブの起動時には計算機の空き資源がジョブを実行するに十分と判断されているにもかかわらず、時間の経過に伴い計算機上で実行されている複数のジョブによって資源が消費され、計算機を資源の不足により停止させてしまう可能性がある。

【0005】 なお、計算機の資源が予め設定したしきい値を超えて消費された場合、アラーム等によりオペレータに通知し、オペレータがジョブのキャンセル等の措置を行うことにより、計算機の停止を未然に防ぐことは可能である。しかし、実行を中止したジョブを他の計算機で再起動する必要があること、及び計算機が終日運転する場合、オペレータが常時監視しなければならないという不便さがある。これは、ジョブ配分の判断が、計算機資源のジョブ起動時点における情報のみによって行われていることに理由があり、これを解決するためには、計算機上で実行されているすべてのジョブがそのライフタイム（ジョブの起動から停止までの時間）にわたって消費する資源を管理する必要がある。

【0006】 本発明の目的は、ノードの稼働停止を防止することのできる分散計算機システムを提供することにある。

【0007】

【課題を解決するための手段】 上記の目的を達成するために、本発明は、複数の計算機から構成された分散計算機システムにおいて、実行対象のジョブの消費予定資源を予め定義するジョブ消費予定資源定義手段と、前記ジョブの実行時に、そのジョブを実行するノード上で必要とされる将来の消費予定資源が前記ノードの有する資源を超えない様にジョブの配分を行うジョブ配分手段とを備えた構成にしている。この構成によれば、予め各ジョブの使用予定資源をジョブ消費予定資源定義手段に定義しておき、ジョブの実行時に、個々のノードに一度にジョブが配分されないような配分、つまりジョブ配分を各ノードの有する資源を超えないように決定すれば、ジョブの起動に際して該ジョブを実行するために必要な計算

機上の資源が、ジョブのライフタイムにわたって予測消費資源として仮想的に確保される。この結果、計算機上で実行されている全てのジョブがそのライフタイムにわたって消費する資源を管理できるようになり、計算機資源が各ジョブの実行に伴って不足する事態を招くことがなくなる。

【0008】

【発明の実施の形態】以下、本発明の実施の形態について説明する。図1は本発明による分散計算機システムのジョブ配分処理部分を示すブロックを示している。図1には分散計算機システムのジョブ制御ノード1と、このジョブ制御ノード1に接続されたジョブ実行ノード2が示されている。ジョブ制御ノード1は、ユーザからのジョブ実行要求を基に動作するジョブ配分制御部11、定義されたジョブ消費予定資源をジョブ配分制御部11に提供するジョブ消費予定資源定義表12、予め定義された各実行ノードの保有資源情報を保持し、これをジョブ配分制御部11に提供する実行ノード保有資源定義表13、ノード上の予測消費資源をジョブ配分制御部11に提供する予測消費資源統括管理表14、及びジョブ実行ノード2側から予測消費資源情報を収集して予測消費資源統括管理表14に反映する予測消費資源収集部15を備えている。

【0009】また、ジョブ実行ノード2は、ジョブ制御ノード1側からの実行要求により起動するジョブ実行制御部21、定義されたジョブ消費予定資源をジョブ実行制御部21に提供するジョブ消費予定資源定義表22、ジョブ実行制御部21により当該実行ノードの予測消費資源が更新される予測消費資源管理表23、及び予測消費資源管理表23の内容を参照して予測消費資源を管理する予測消費資源管理部24を備えている。なお、ジョブ実行ノード2は複数が存在可能であるが、図1では1つのみの構成を示し、その動作説明においても1つのみについて説明する。また、図1においては、ジョブ制御ノード1とジョブ実行ノード2を別個のノードとして構成しているが、ジョブ制御ノード1がジョブ実行ノード2を含む構成であってもよい。

【0010】以上の構成において、ジョブ配分制御部11は、ユーザからのジョブ実行要求に対し、予測消費資源統括管理表14で管理される予測消費資源（ジョブ実行ノード上の） R_{fc} と、ジョブ消費予定資源定義表12に定義されるジョブの消費予定資源 R_{cp} の和（ $R_{fc} + R_{cp}$ ）に対する前記 R_{fc} の比率 $R_{fc} / (R_{fc} + R_{cp})$ が最小になり、且つ、前記 $R_{fc} + R_{cp}$ が実行ノード保有資源定義表13に定義される当該ジョブ実行ノードの保有資源を超えないジョブ実行ノード2を選択する。この選択したジョブ実行ノード2上のジョブ実行制御部21に対し、ジョブの実行要求が行われる。

【0011】ジョブ配分制御部11からジョブの実行要求を受けたジョブ実行制御部21は、そのジョブを起動

すると共にジョブ消費予定資源定義表22から前記ジョブの消費予定資源値を取得する。この消費予定資源値は、ジョブ実行制御部21によって予測消費資源管理表23に加算され、また、ジョブの実行が終了すれば、前記ジョブの消費予定資源値は予測消費資源管理表23から減算され、予測消費資源の更新が行われ、予測消費資源の状態が管理される。この後、予測消費資源管理部24によって予測消費資源管理表23の参照が行われ、この参照結果を基に予測消費資源情報が取得され、この予測消費資源情報は予測消費資源管理部24から予測消費資源収集部15へ送出される。

【0012】このように、予め各ジョブの使用予定資源をジョブ消費予定資源定義表12、22に定義しておき、ジョブの実行時に、各ノードにジョブが一度に配分されないようにジョブ配分制御部11によって決定すれば、計算機資源を有効に活用できる。この結果、計算機上で実行されている全てのジョブがそのライフタイムにわたって消費する資源を管理でき、計算機資源が各ジョブの実行に伴って不足する事態は生じなくなる。

【0013】本発明は、特に、画像処理や科学技術計算分野のように大量の資源を消費する類似のジョブを多数実行する等の用途に有効である。つまり、各計算機に一度にジョブを配分し過ぎることによって計算機に許容以上の資源が要求され、計算機の稼働に悪影響を与えるようなシステムへの適用に有効である。

【0014】図2は図1に示したジョブ配分制御部11の詳細を示すブロック図である。ジョブ配分制御部11は、ジョブ実行要求管理部16、ジョブ実行要求キュー管理表17、及びジョブ配分処理部18を備えて構成されている。ジョブ実行要求管理部16は、ユーザからジョブ実行要求を受けたときにジョブ実行要求をジョブ実行要求キュー管理表17のキューに登録する。また、ジョブ配分処理部18は、予め定義された周期毎、又はユーザからジョブ実行要求があった時ジョブ実行要求管理部16からの起動指令に基づいて、ジョブ実行要求キュー管理表17からジョブ実行要求を取り出し、ジョブ消費予定資源定義表12、実行ノード保有資源定義表13及び予測消費資源統括管理表14の3つを参照して各ジョブのジョブ実行ノードへの配分を実行する。

【0015】更に、ジョブ配分処理部18は、どのジョブ実行ノード上に対しても必要とする予測消費資源を確保できなかったジョブの配分を見合わせ、ジョブ実行要求キュー管理表17内の次の順番のジョブを配分することを試みる。これは、ジョブ実行ノードの効率的な使用を実現するものである。しかし、このようなジョブの配分を行っていると、大量の消費予定資源を必要とするジョブが、いつまでもジョブ実行ノード2に配分されないケースが発生する。そこで、ジョブ配分処理部18は、ジョブの配分をスキップする度にジョブ実行要求キュー管理表17内の当該ジョブのスキップカウンタ（図示せ

ず)をインクリメントする。そして、カウント値が予め定義(スキップカウンタに)したしきい値に達した場合、前記ジョブが優先的に実行されるように処理する。このジョブ配分方式にすることで、ジョブに対する公平な資源配分の実現、すなわちジョブを公平に実行することが可能になる。なお、ジョブ消費予定資源定義表22は、ジョブ配分制御部11がジョブ実行制御部21に対してジョブの実行要求を行う時に、当該ジョブの消費予定資源を合わせて通知することにより削除することも可能である。

【0016】次に、本発明による分散計算機システム的具体例について説明する。図3は、大量の入力データに処理及び加工を施して出力するジョブを複数のデータ処理サーバ上で実行する分散計算機システムのブロック図を示している。入力データのサイズは、ユーザから与えられるジョブ実行要求毎に異なるので、各ジョブが出力するデータのサイズもジョブ実行要求毎に異なる。このため、従来のジョブ配分技術では、データ処理サーバ上で各ジョブの実行が進むにつれ、出力データを格納するディスク(記憶媒体)を使いきってしまう可能性があった。ディスクに出力データを格納できないということは、データ処理サーバの稼働停止を意味する。したがって、かかる事態はジョブ配分の過程で避ける必要があった。しかし、本発明の適用により、各データ処理サーバ上で実行中のジョブがそのライフタイムにわたって消費予定のディスク容量を把握でき、その情報に基づいたジョブ配分を行うことが可能になるので、データ処理サーバの稼働停止の危険性を著しく低下させることができる。

【0017】以下、図3の構成及び動作について説明する。図3には管理エージェント101と、複数のデータ処理サーバ201の構成が示され、管理エージェント101は図1のジョブ制御ノード1に相当し、データ処理サーバ201は図1のジョブ実行ノード2に相当する。管理エージェント101は、ジョブ属性定義ファイル121、ジョブ属性ジェネレータ122、図1のジョブ消費予定資源定義表12に相当するジョブ属性定義テーブル123、ノード記憶資源定義ファイル131、ノード記憶資源ジェネレータ132、図1の実行ノード保有資源管理表13に相当するノード記憶資源定義テーブル133、記憶資源統括アカウントテーブル141、記憶資源アカウント収集マスタ151、図2のジョブ実行要求管理部16に相当するジョブ実行受付処理部161、図2のジョブ実行要求キュー管理表17に相当するジョブ管理テーブル171、図2のジョブ配分処理部18に相当するジョブディスパッチャ181、及びジョブ状態更新処理部182を備えて構成されている。

【0018】また、データ処理サーバ201は、図1のジョブ実行制御部21に相当するジョブ起動処理部211、ジョブシェル212、記憶資源アカウントテーブル

231、及び図1の予測消費資源管理部24に相当する記憶資源アカウント収集スレーブ241を備えて構成されている。なお、データ処理サーバ201は複数台の存在が可能であるが、以下においてはデータ処理サーバ201の1つについてのみ説明する。

【0019】管理エージェント101において、ジョブ属性定義テーブル123は管理エージェントの不図示のメモリ上に存在しており、これに対してジョブ属性定義ファイル121及びジョブ属性ジェネレータ122はその補助機構として機能する。具体的に説明すると、ユーザがジョブ属性定義ファイル121にジョブの属性を記述し、ジョブ属性ジェネレータ122を実行することによって、ジョブ属性定義ファイル121に格納したジョブの属性をメモリ上のジョブ属性定義テーブル123に反映することができる。本実施例において、ジョブの属性とは、(1)ジョブ実行要求時にユーザから与えられる入力データのサイズに対してジョブ実行の過程で生成される中間データ及び最終的な出力データのサイズの関係式、(2)上記ジョブを起動する場合のスキップリミット値、を含むものである。

【0020】ノード記憶資源定義テーブル133は管理エージェント101のメモリ上に存在し、これに対してノード記憶資源定義ファイル131及びノード記憶資源ジェネレータ132は、ノード記憶資源定義テーブル133の補助機構として機能する。具体的に説明すると、ユーザによりノード記憶資源定義ファイル131に各ノードが有する記憶資源量の情報が記述され、ノード記憶資源ジェネレータ132が動作することにより、各ノードが有する記憶資源量をメモリ上のノード記憶資源定義テーブル133に反映することができる。本実施例において、ノード記憶資源定義ファイル131に定義されるノード記憶資源とは、各データ処理サーバが有するディスクの容量を含むものである。

【0021】ジョブ管理テーブル171は、管理エージェント101のメモリ上に存在する。また、ジョブディスパッチャ181は予め定義された周期で起動し、各データ処理サーバ201の記憶資源アカウント情報テーブル231の内容の収集を記憶資源アカウント収集マスタ151に要求し、その収集結果が格納される記憶資源統括アカウントテーブル141を参照してジョブ実行要求をジョブ管理テーブル171から取り出し、そのジョブをデータ処理サーバ201のいずれかにディスパッチする。

【0022】ここで、ジョブディスパッチャ181は、ジョブを実行するために必要な記憶資源の空きがデータ処理サーバ201のいずれにも存在しない場合、そのジョブの実行を見合わせ、ジョブ管理テーブル171内のジョブのスキップカウンタをインクリメントする。スキップカウンタの値がジョブ属性定義テーブル123から与えられるスキップリミット値に達した場合、以後ジョ

ブディスパッチャ181はそのジョブをスキップさせず、データ処理サーバ201のいずれかにおいてジョブを実行するのに十分な記憶資源の空きが生じるまで待機する。また、ジョブディスパッチャ181はジョブのディスパッチに際し、ジョブ属性定義テーブル123から取得した情報を基にジョブディスパッチャ181が生成するジョブの記憶資源アカウント値をジョブ起動パラメータとし、これをジョブ起動処理部211へ転送する。これにより、図1で必要であったジョブ消費予定資源定義表22を省略することができる。

【0023】ジョブ管理テーブル171は、ユーザから要求されたジョブのキュー管理情報に限らず、データ処理サーバ201上で実行されている各ジョブの状態も、ジョブ要求受付処理部161を通して保持する。なお、図3には示していないが、ジョブ管理テーブル171は、画面等を通して各ジョブの実行状態をユーザに提供するための元データとして用いることができる。

【0024】記憶資源統括アカウントテーブル141は図1の予測消費資源統括管理表14に相当し、管理エージェントのメモリ上に存在する。また、記憶資源アカウント収集マスタ151は図1の予測消費資源収集部15に相当する。この記憶資源アカウント収集マスタ151は、ジョブディスパッチャ181からの要求に従って記憶資源アカウント収集スレーブ241から各データ処理サーバ201の記憶資源アカウント情報を収集し、記憶資源統括アカウントテーブル141へ格納する。記憶資源アカウントテーブル231は図1の予測消費資源管理表23に相当し、データ処理サーバのメモリ上に存在している。また、記憶資源アカウント収集スレーブ241は記憶資源アカウント収集マスタ151からの要求により、記憶資源アカウント収集テーブル231の内容を記憶資源アカウント収集マスタ151へ転送する。ジョブシェル212はジョブを包含するものであり、図1に示したジョブ実行制御部21が予測消費資源管理表23へ予測消費資源を格納する部分の機能と同一の機能を有している。

【0025】図4はジョブシェル212で実行される処理を示すフローチャートである。まず、ジョブディスパッチャ181からジョブ起動処理部211に対してジョブ起動要求のパラメータとして渡された記憶資源アカウント値を取得する（ステップS401）。次に、取得した記憶資源アカウント値を記憶資源アカウントテーブル231に加算する（ステップS402）。更に、ジョブが実行され（ステップS403）、このジョブの終了後、記憶資源アカウントテーブル231から記憶資源アカウント値を減算する（ステップS404）。最後に、ジョブ状態更新処理部182に対してジョブの終了通知を実行する（ステップS405）。このように、簡単な処理により、ジョブの実行に伴い記憶資源統括アカウントテーブル141の内容を更新することができる。

【0026】図5は、ジョブ属性定義テーブル123の構成内容を示している。ここではジョブ毎に、消費予定メモリ容量、消費予定ディスク容量及びスキップリミットを定義している。図5において、ジョブ「JOB-A」の消費予定ディスク容量は、固定的に使用される50MBに、入力データサイズの2.2倍を加えて算出（ $50 + 2.2 \times \text{入力データサイズ (MB)}$ ）される。

【0027】図6はジョブ管理テーブル171の構成内容を示している。図6において、第1エントリのジョブ「A01」は、消費予定メモリ容量及び消費予定ディスク容量を確保できるデータ処理ホストが現状存在しないために、起動がスキップされていることを示している。この例では、スキップカウンタがスキップリミットに達しているため、ジョブディスパッチャ181は、これ以上ジョブ「A01」の起動をスキップせず、ジョブ「A01」の実行に必要な消費予定メモリ容量及び消費予定ディスク容量が、データ処理ホストのいずれかに確保できるまで他のジョブの起動要求は行わない。

【0028】図7はノード記憶資源定義テーブル133の構成内容を示している。ここでは、データ処理サーバ毎に、スワップ容量を含めた保有メモリ容量、及び保有ディスク容量を定義している。図8は記憶資源アカウントテーブル231の構成内容を示している。本例では、当該データ処理ホスト上で実行されている各ジョブの消費予定メモリ容量及び消費予定ディスク容量が格納されている。

【0029】図9は記憶資源統括アカウントテーブル141の構成内容を示している。この例では、データ処理サーバ毎に、ジョブの消費予定メモリ容量合計及び消費予定ディスク容量合計を格納している。以上のように、データ処理サーバ201を備えたシステムにおいても、予め各ジョブの使用予定資源をジョブ属性定義テーブル123及びデータ処理サーバ201に定義しておくことにより、計算機上で実行されている全てのジョブがそのライフタイムにわたって消費する資源を管理できるようになり、計算機資源が各ジョブの実行に伴って不足する事態は生じない。

【0030】図10は図3の構成を発展させた本発明の他の具体例を示すブロック図である。図10においては、管理エージェント101及びデータ処理サーバ201で共有できる情報が、データベースサーバ301で管理されるシステム構成が示されている。データベースサーバ301上における情報の共有は、NFS（網ファイルシステム：Network File/System）等によるファイルの共有、SQL（構造化照会言語：Structured Query Language）等を使用したDBMS（データベース管理システム：Data Base Management System）等によるものである。

【0031】図10の分散計算機システムの構成について説明すると、図1のジョブ制御ノードに相当する管理

エージェント101、複数のデータ処理サーバ201、及び管理エージェント101とデータ処理サーバ201に接続されるデータベースサーバ301を備えて構成されている。ここでは、三者を独立した形で構成しているが、データベースサーバ301が管理エージェント101又はデータ処理サーバ201を兼ねた構成にすることもできる。

【0032】管理エージェント101は、図2のジョブ実行要求管理部16に相当するジョブ実行受付処理部161と図2のジョブ配分処理部18に相当するジョブディスパッチャ181を備えて構成されている。また、データ処理サーバ201は、図1のジョブ実行制御部21に相当するジョブ起動処理部211、及びジョブシェル212を備えて構成されている。また、データベースサーバ301は、図2のジョブ実行要求キュー管理表17に相当するジョブ管理テーブル311、図1のジョブ消費予定資源定義表12に相当するジョブ属性定義テーブル312、図1の実行ノード保有資源管理表13に相当するノード記憶資源定義テーブル313、及び図1の予測消費資源統括管理表14に相当する記憶資源統括アカウントテーブル314を備えて構成されている。

【0033】管理エージェント101のジョブ実行受付処理部161は、ユーザからのジョブ実行要求を受け付け、データベースサーバ301内のジョブ管理テーブル311にジョブ実行要求のエントリを追加する。ジョブディスパッチャ181は予め定義された周期で起動し、ジョブ管理テーブル311からジョブ実行要求を取り出し、ジョブ属性定義テーブル312に定義されたジョブの予測消費資源情報、ノード記憶資源定義テーブル313に定義された各データ処理サーバの保有記憶資源情報、及び記憶資源統括アカウントテーブル314で管理されている各データ処理サーバの記憶資源消費情報の各々に基づいて、ジョブを実行するデータ処理サーバを決定する。更に、ジョブディスパッチャ181は、ジョブ属性定義テーブル312に定義される情報に基づいて算出したジョブの記憶資源アカウント値をパラメータとして、ジョブ起動処理部211へジョブの起動要求を送出する。この要求に従って、ジョブ起動処理部211はジョブシェル212を起動させる。このジョブシェル212の処理について、以下に説明する。

【0034】図11はジョブシェルの処理を示すフローチャートである。まず、ジョブディスパッチャ181からジョブ起動処理部211へジョブ起動要求のパラメータとして渡された記憶資源アカウント値を取得する（ステップS601）。次に、この記憶資源アカウント値を記憶資源統括アカウントテーブル314に加算する（ステップS602）。更に、起動させたジョブを実行し（ステップS603）、このジョブの実行が終了した後、記憶資源統括アカウントテーブル314の記憶資源アカウント値を減算する（ステップS604）。ついで、

ジョブ管理テーブル311にジョブの終了ステータスを反映させる（ステップ605）。図11の処理により、ジョブの実行に伴い記憶資源統括アカウントテーブル314の内容を更新することができる。

【0035】以上のように、図10の構成によれば、管理エージェント101及びデータ処理サーバ201で共有できる情報がデータベースサーバ301で管理されるシステムにおいても、図3の場合と同様に、予め各ジョブの使用予定資源をジョブ属性定義テーブル312に定義し、これを用いて適正配分を行うことにより、ジョブの起動に際して該ジョブを実行するために必要な計算機上の資源をジョブのライフタイムにわたって予測消費資源として仮想的に確保でき、計算機上で実行されている全てのジョブがそのライフタイムにわたって消費する資源を管理できるようになり、計算機資源が各ジョブの実行に伴って不足する事態は生じない。

【0036】

【発明の効果】以上説明した通り、本発明によれば、計算機システムにおいて、実行するジョブの消費予定資源を予めジョブ属性として定義しておくことにより、ジョブの起動時に、ノード上に必要とされる将来の消費予定資源が、ノードが有している資源を超えないようなジョブの配分が行え、ジョブ実行ノードの稼働停止の発生を著しく低減した分散計算機システムを実現することができる。

【図面の簡単な説明】

【図1】本発明による分散計算機システムのジョブ配分処理部分を示すブロック図である。

【図2】図1のジョブ配分制御部の詳細を示すブロック図である。

【図3】本発明の第1の具体例の構成を示すブロック図である。

【図4】図3のジョブシェルの処理を説明するフローチャートである。

【図5】ジョブ属性定義テーブル123の構成例図である。

【図6】ジョブ管理テーブル171の構成例図である。

【図7】ノード記憶資源定義テーブル133の構成例図である。

【図8】記憶資源アカウントテーブル231の構成例図である。

【図9】記憶資源統括アカウントテーブル141の構成例図である。

【図10】本発明の第2の具体例の構成を示すブロック図である。

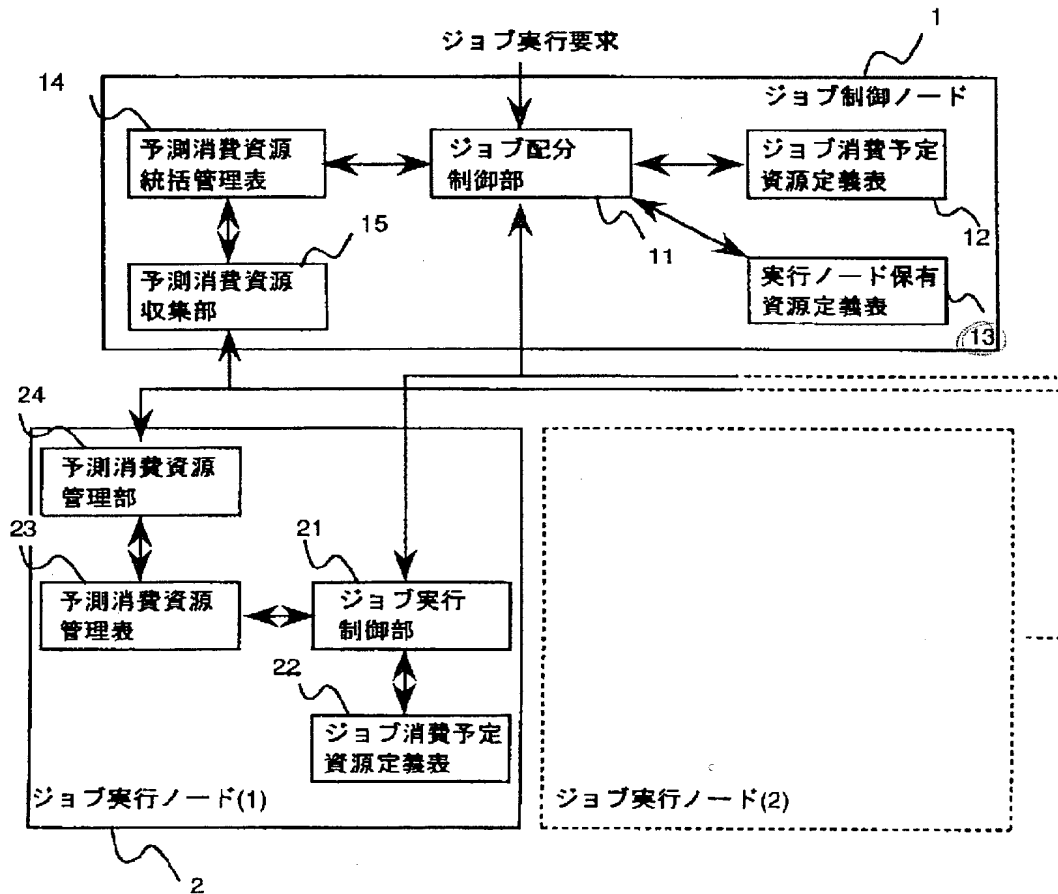
【図11】図10のジョブシェルの処理を説明するフローチャートである。

【符号の説明】

- 1 ジョブ制御ノード
- 2 ジョブ実行ノード

- | | | | |
|----|---------------|---------|-----------------|
| 11 | ジョブ配分制御部 | 24 | 予測消費資源管理部 |
| 12 | ジョブ消費予定資源定義表 | 101 | 管理エージェント |
| 13 | 実行ノード保有資源管理表 | 123、312 | ジョブ属性定義テーブル |
| 14 | 予測消費資源統括管理表 | 133、313 | ノード記憶資源定義テーブル |
| 15 | 予測消費資源収集部 | 141、314 | 記憶資源統括アカウントテーブル |
| 16 | ジョブ実行要求管理部 | 181 | ジョブディスパッチャ |
| 17 | ジョブ実行要求キュー管理表 | 201 | データ処理サーバ |
| 18 | ジョブ配分処理部 | 211 | ジョブ起動処理部 |
| 21 | ジョブ実行制御部 | 212 | ジョブシェル |
| 22 | ジョブ消費予定資源定義表 | 301 | データベースサーバ |
| 23 | 予測消費資源管理表 | | |

【図1】

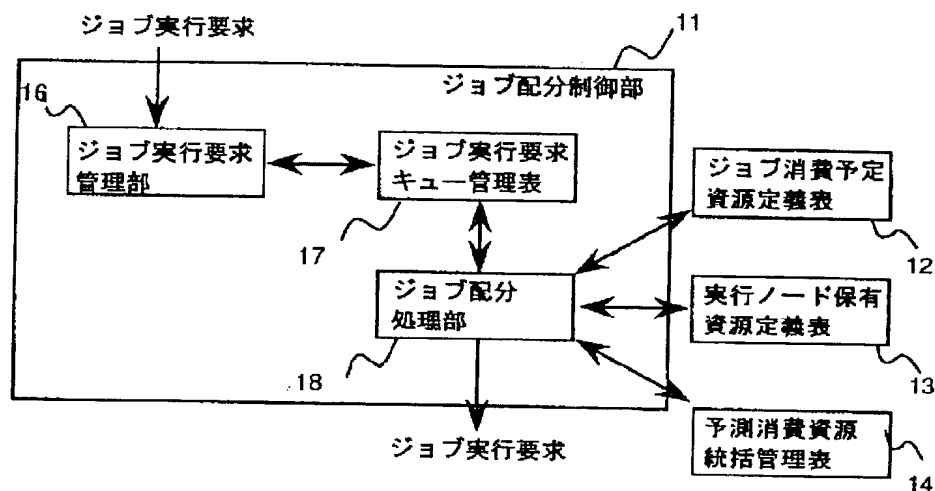


【図8】

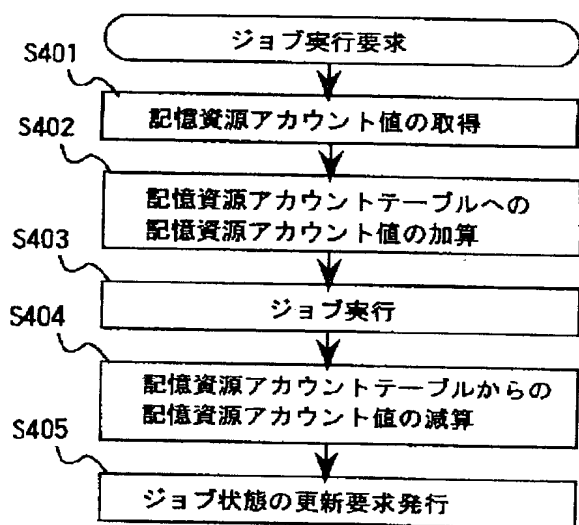
テーブル231

ジョブID	消費予定メモリ容量 (MB)	消費予定ディスク容量 (MB)
B01	5	30

【図2】



【図4】



【図7】

テーブル133

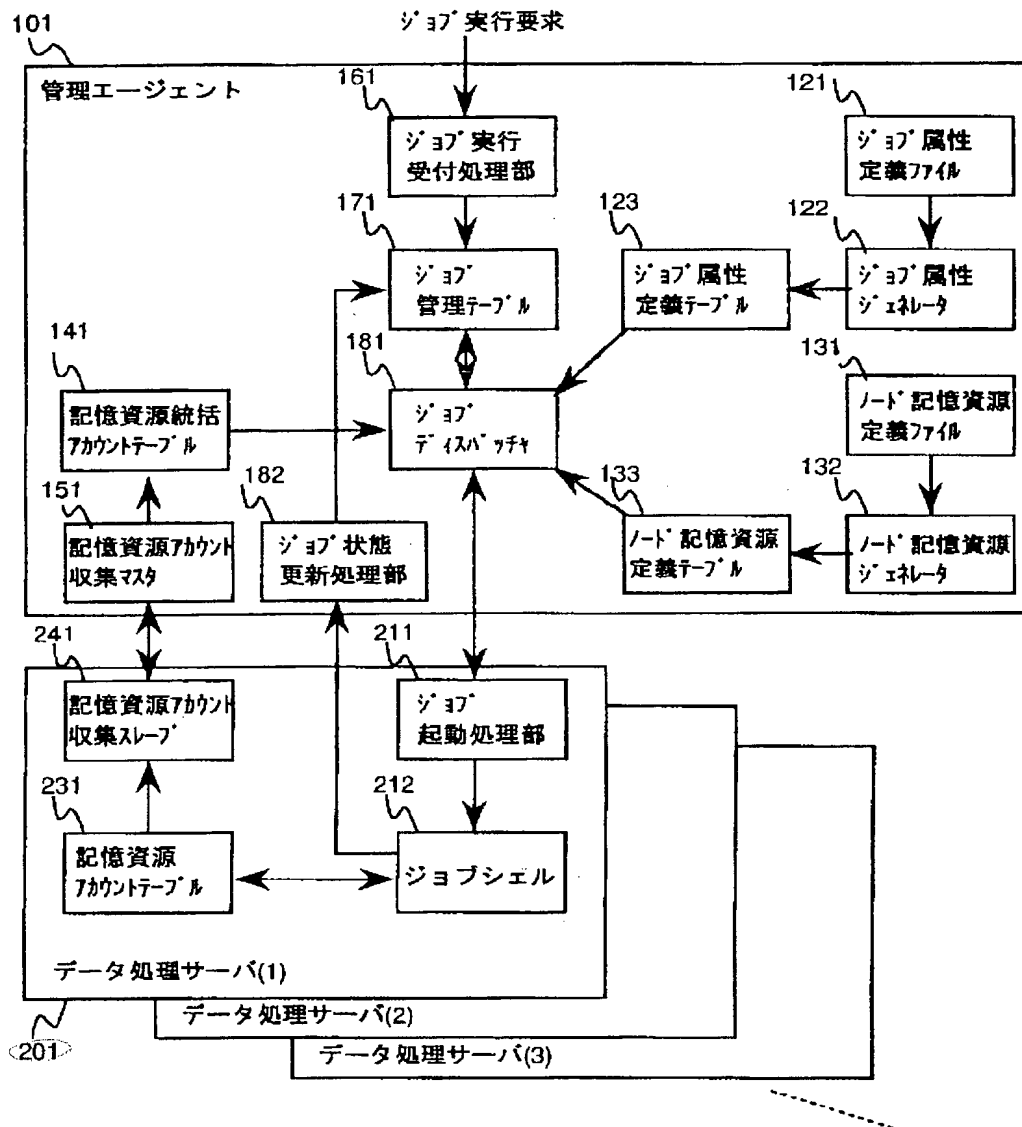
ノードID	保有メモリ (MB)	保有ディスク容量 (MB)
SRV01	2,048	30,000
SRV02	1,024	20,000

【図5】

テーブル123

ジョブ名称	消費予定メモリ容量 (MB)	消費予定ディスク容量 (MB)	スキップリミット
JOB-A	180	50 + 2.2 × 入力データサイズ	5
JOB-B	5	30	2

【図3】



【図6】

テーブル171

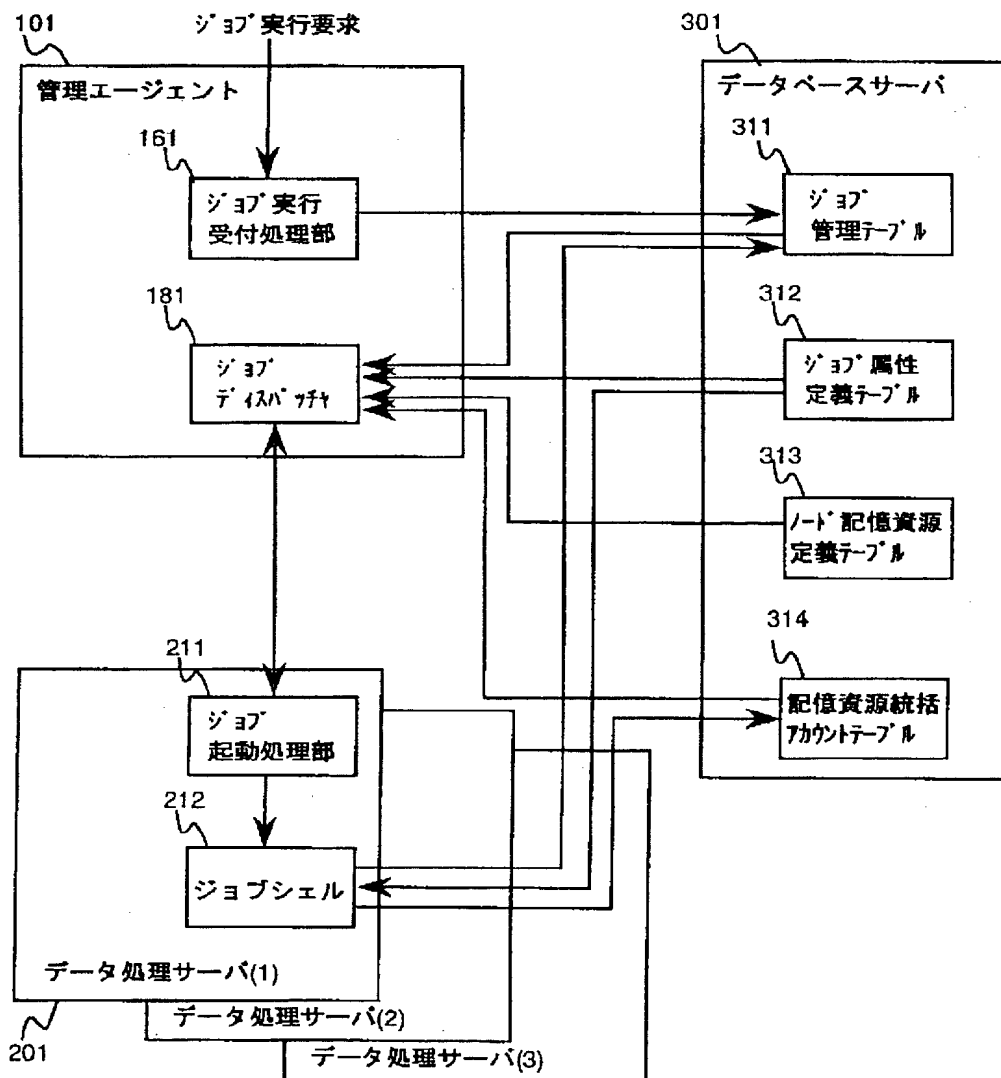
ジョブID	ジョブ名称	消費予定メモリ(MB)	消費予定ディスク容量(MB)	スキップカウンタ	スキップカウンタ	ジョブ状態
A01	JOB-A	180	11,050	5	5	起動待ち
B01	JOB-B	5	30	0	3	実行中
A01	JOB-B	180	3,800	1	5	起動待ち

【図9】

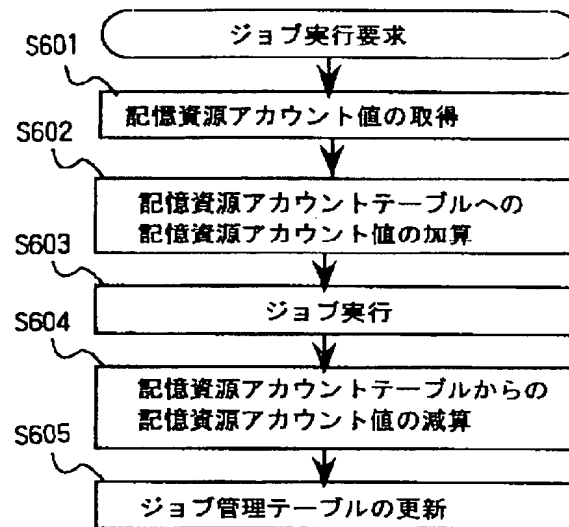
テーブル141

ノードID	消費予定メモリ容量 (MB)	消費予定ディスク容量 (MB)
SRV01	800	25,000
SRV02	500	15,000

【図10】



【図11】



フロントページの続き

(72) 発明者 田中 紀夫
茨城県日立市大みか町五丁目2番1号 株
式会社日立製作所大みか工場内